Лабораторная работа №9

Тема: Построение кривых на плоскости

Задание:

1. Создать функцию

double Lagr(CMatrix &X, CMatrix &Y, double x)

которая по множеству точек на плоскости (X, Y)

$$X = (x_0 \quad x_1 \quad \dots \quad x_N)^T, \quad Y = (y_0 \quad y_1 \quad \dots \quad y_N)^T$$

вычисляет значение интерполяционного **полинома Лагранжа** в точке $x \in [x_0; x_N]$.

2. Пункт меню **«Lines ► Lagr».**

Получить множество точек

$$X = (x_0 \quad x_1 \quad \dots \quad x_N)^T, \quad Y = (y_0 \quad y_1 \quad \dots \quad y_N)^T$$

$$y_i = f(x_i), \quad x \in [0, \pi], \quad \Delta x = \frac{\pi}{4}, \quad f(x) = [2 + \cos(x)]^{\sin(2x)}$$

По полученному набору данных рассчитать значения полинома Лагранжа $L(x)$ на отрезке $x \in [x_0, x_N]$ с шагом $\Delta x = 0{,}2$ и построить его график.

Для построения графика использовать класс CPlot2D.

3. Создать функцию

void Bezier(CMatrix &X, CMatrix &Y, CMatrix &XB, CMatrix &YB, int M);

которая по множеству точек на плоскости

$$X = (x_0 \quad x_1 \quad \dots \quad x_N)^T, \quad Y = (y_0 \quad y_1 \quad \dots \quad y_N)^T$$

вычисляет координаты **кривой Безье**

$$XB = (x_0^b \quad x_1^b \quad \dots \quad x_M^b)^T, \quad YB = (y_0^b \quad y_1^b \quad \dots \quad y_M^b)^T,$$

M – число отрезков, на которые разбивается параметр *t*, $t \in [0, 1]$

Для вычисления координат кривой Безье использовать геометрический алгоритм.

4. Пункт меню **«Lines ► Bezier».**

Построить варианты кривых Безье (из учебника, рис. 14.5). Набор данных (в мировой системе координат) для каждой кривой задать вручную в обработчике пункта меню.

Для построения кривых использовать класс CPlot2D.

Теоретические основы:

В инженерных расчетах часто требуется установить функцию *f(x)* для всех значений ***x*** отрезка **[***a,b***]** , если известны ее значения в некотором конечном числе точек этого отрезка. Одним из способов приближения функции является *интерполяция.*

Интерполяцио́нный многочле́н Лагра́нжа — многочлен минимальной степени, принимающий данные значения в данном наборе точек. Для $n+1$ пар чисел (x_0, y_0), $(x_1, y_1),\ldots, (x_n, y_n)$, где все x_j различны, существует единственный многочлен $L(x)$ степени не более n, для которого $L(x_j) = y_j$.

В простейшем случае ($n=1$) — это линейный многочлен, график которого — прямая, проходящая через две заданные точки.

Ход работы:

ChildView – создание меню и обработчик событий

MainFrm – создаем окно

LibGraph – нахождение координат точек

CMatrix –реализовываем функции для создания матриц и расчета координат наших функций, файлы ChildView.h и CMatrix.h. И осуществляем вывод этих функций в окно приложения.

Для выполнения первого задания данной лабораторной работы необходимо реализовать функцию *double Lagr(CMatrix &X, CMatrix &Y, double x)* в файле LibGraph.cpp:

```
double Lagr(CMatrix& X, CMatrix& Y, double x, int size)
{
 double lagrange_pol = 0;
 double basics_pol;

 for (int i = 0; i < size; i++)
 {
        basics_pol = 1;
        for (int j = 0; j < size; j++)
        {
               if (j == i)
                      continue;
               basics_pol *= (x - X(j)) / (X(i) - X(j));
        }
        lagrange_pol += basics_pol * Y(i);
 }
 return lagrange_pol;
}
```

Для выполнения второго задания данной лабораторной необходимо в файле ChildView.cpp получить множество точек

$$X = (x_0 \quad x_1 \quad \dots \quad x_N)^T, \quad Y = (y_0 \quad y_1 \quad \dots \quad y_N)^T$$

$$y_i = f(x_i), \quad x \in [0, \pi], \quad \Delta x = \frac{\pi}{4}, \quad f(x) = [2 + \cos(x)]^{\sin(2x)}$$

И по полученному набору данных рассчитать значения полинома Лагранжа $L(x)$ на отрезке $x \in [x_0, x_N]$ с шагом $\Delta x = 0{,}2$:

```
void CChildView::OnLagr()
{
 double dx = pi / 4;
 double xL = 0;
 double xH = pi;
 int N = (xH - xL) / dx;
 X.RedimMatrix(N + 1);
 Y.RedimMatrix(N + 1);
 for (int i = 0; i <= N; i++)
 {
        X(i) = xL + i * dx;
        Y(i) = pow(2 + cos(X(i)), sin(2 * X(i)));
 }
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 1;
 Invalidate();
}
```

Для построения графика использовуем класс CPlot2D. Создаем и определяем наши поля и прототипы функций в LibGraph.h:

```
class CPlot2D
{
 CMatrix X;                          // Аргумент
 CMatrix Y;                          // Функция
 CMatrix K;                          // Матрица пересчета координат
 CRect RW;                                  // Прямоугольник в окне
 CRectD RS;                          // Прямоугольник области в МСК
 CMyPen PenLine;   // Перо для линий
 CMyPen PenAxis;   // Перо для осей
public:

 CPlot2D() { K.RedimMatrix(3, 3); };
 void SetParams(CMatrix& XX, CMatrix& YY, CRect& RWX);
 void GetWindowCoords(double xs, double ys, int &xw, int &yw);
 void SetPenLine(CMyPen& PLine);   // Перо для рисования графика
 void SetPenAxis(CMyPen& PAxis);   // Перо для осей координат
 void Draw(CDC& dc, int Ind1, int Int2);        // Рисование с самостоятельным пересчетом координат
 void DrawBezier(CDC& dc, int NT);
 void DrawLagr(CDC& dc);
};
```

Реализация в LibGraph.cpp:

```
void CPlot2D::DrawLagr(CDC& dc)
{
 double dx = pi / 4;
 double xL = 0;
 double xH = pi;
 int N = (xH - xL) / dx;
```

```
 dx = 0.2;
 int NL = (xH - xL) / dx;
 CMatrix XL(NL + 1);
 CMatrix YL(NL + 1);

 for (int i = 0; i <= NL; i++)
 {
        XL(i) = xL + i * dx;
        YL(i) = Lagr(X, Y, XL(i), N + 1);
 }

 double xs, ys;
 int xw, yw;
 xs = XL(0); ys = YL(0);
 GetWindowCoords(xs, ys, xw, yw);                // координаты начальной точки графика в ОСК
 CPen MyPen(PenLine.PenStyle, PenLine.PenWidth, PenLine.PenColor);
 CPen* pOldPen = dc.SelectObject(&MyPen);
 dc.MoveTo(xw, yw);                                          // Перо в начальную точка для рисования
графика
 for (int i = 1; i < XL.rows(); i++)
 {
        xs = XL(i); ys = YL(i);
        GetWindowCoords(xs, ys, xw, yw);         // координаты начальной точки графика с номером i в ОСК
        dc.LineTo(xw, yw);
 }
 dc.SelectObject(pOldPen);
}

void CPlot2D::GetWindowCoords(double xs, double ys, int &xw, int &yw)
// Пересчитывает координаты точки из МСК в оконную
// xs - x- кордината точки в МСК
// ys - y- кордината точки в МСК
// xw - x- кордината точки в оконной СК
// yw - y- кордината точки в оконной СК

{
 CMatrix V(3), W(3);
 V(2) = 1;
 V(0) = xs;
 V(1) = ys;
 W = K * V;
 xw = (int)W(0);
 yw = (int)W(1);
}

void CPlot2D::SetPenAxis(CMyPen& PAxis)
// Установка параметров пера для линий осей
{
 PenAxis.PenStyle = PAxis.PenStyle;
 PenAxis.PenWidth = PAxis.PenWidth;
 PenAxis.PenColor = PAxis.PenColor;
}

void CPlot2D::Draw(CDC& dc, int Ind1, int Ind2)
// Рисует график в режиме MM_TEXT - собственный пересчет координат
// dc - ссылка на класс CDC MFC
// Ind1=1/0 - рисовать/не рисовать рамку
// Ind2=1/0 - рисовать/не рисовать оси координат
{
 double xs, ys;  // мировые  координаты точки
 int xw, yw;     // оконные координаты точки
 if (Ind1 == 1)dc.Rectangle(RW);                                                    // Рамка в окне
```

```
 if (Ind2 == 1)      // Если нужны оси...
 {//***
        CPen MyPen(PenAxis.PenStyle, PenAxis.PenWidth, PenAxis.PenColor);
        CPen* pOldPen = dc.SelectObject(&MyPen);
        if (RS.left*RS.right < 0)                                          // Нужна Ось Y
        {
               xs = 0;  ys = RS.top;           // Точка (0,y_max) в МСК
               GetWindowCoords(xs, ys, xw, yw); // (xw,yw) -точка (0,y_max) в ОСК
               dc.MoveTo(xw, yw);               // Перо в точку (0,y_max)

               xs = 0;  ys = RS.bottom;        // Точка (0,y_min) в МСК
               GetWindowCoords(xs, ys, xw, yw); // (xw,yw) -точка (0,y_min) в ОСК
               dc.LineTo(xw, yw);               // Линия (0,y_max) - (0,y_min) - Ось Y
        }

        if (RS.top*RS.bottom < 0)                                          // Нужна Ось X
        {
               xs = RS.left;  ys = 0;           // (xs,ys) - точка (x_min,0) в МСК
               GetWindowCoords(xs, ys, xw, yw); // (xw,yw) -точка (x_min,0) в ОСК
               dc.MoveTo(xw, yw);                  // Перо в точку (x_min,0)

               xs = RS.right;  ys = 0;          // (xs,ys) - точка (x_max,0) в МСК
               GetWindowCoords(xs, ys, xw, yw); // (xw,yw) -точка (x_max,0) в ОСК
               dc.LineTo(xw, yw);                  // Линия (x_min,0) - (x_max,0) - Ось X
        }
        dc.SelectObject(pOldPen);

 }

 xs = X(0); ys = Y(0);
 GetWindowCoords(xs, ys, xw, yw); // координаты начальной точки графика в ОСК
 CPen MyPen(PenLine.PenStyle, PenLine.PenWidth, PenLine.PenColor);
 CPen* pOldPen = dc.SelectObject(&MyPen);
 dc.MoveTo(xw, yw);        // Перо в начальную точка для рисования графика
 for (int i = 1; i < X.rows(); i++)
 {
        xs = X(i); ys = Y(i);
        GetWindowCoords(xs, ys, xw, yw); // координаты начальной точки графика с номером i в ОСК
        dc.LineTo(xw, yw);
 }
 dc.SelectObject(pOldPen);
}

void CRectD::SetRectD(double l, double t, double r, double b)
{
 left = l;
 top = t;
 right = r;
 bottom = b;
}

CSizeD CRectD::SizeD()
{
 CSizeD cz;
 cz.cx = fabs(right - left);// Ширина прямоугольной области
 cz.cy = fabs(top - bottom);// Высота прямоугольной области
 return cz;
}

CMatrix SpaceToWindow(CRectD& RS, CRect& RW)
// Функция обнавлена
// Возвращает матрицу пересчета координат из мировых в оконные
```

```
// RS - область в мировых координатах - double
// RW - область в оконных координатах - int
{
 CMatrix M(3, 3);
 CSize sz = RW.Size();        // Размер области в ОКНЕ
 int dwx = sz.cx;          // Ширина
 int dwy = sz.cy;          // Высота
 CSizeD szd = RS.SizeD(); // Размер области в МИРОВЫХ координатах

 double dsx = szd.cx;    // Ширина в мировых координатах
 double dsy = szd.cy;    // Высота в мировых координатах

 double kx = (double)dwx / dsx;   // Масштаб по x
 double ky = (double)dwy / dsy;   // Масштаб по y


 M(0, 0) = kx;  M(0, 1) = 0;    M(0, 2) = (double)RW.left - kx * RS.left;                // Обновлено
 M(1, 0) = 0;   M(1, 1) = -ky;  M(1, 2) = (double)RW.bottom + ky * RS.bottom;            // Обновлено
 M(2, 0) = 0;   M(2, 1) = 0;    M(2, 2) = 1;
 return M;
}

void CPlot2D::SetParams(CMatrix& XX, CMatrix& YY, CRect& RWX)
// XX - вектор данных по X
// YY - вектор данных по Y
// RWX - область в окне
{
 int nRowsX = XX.rows();
 int nRowsY = YY.rows();
 X.RedimMatrix(nRowsX);
 Y.RedimMatrix(nRowsY);
 X = XX;
 Y = YY;
 double x_max = X.MaxElement();
 double x_min = X.MinElement();
 double y_max = Y.MaxElement();
 double y_min = Y.MinElement();
 RS.SetRectD(x_min, y_max, x_max, y_min);                    // Область в мировой СК
 RW.SetRect(RWX.left, RWX.top, RWX.right, RWX.bottom); // Область в окне
 K = SpaceToWindow(RS, RW);                                          // Матрица пересчета
координат
                                          }
```

Для третьего задания реализуем функцию DrawLarg в файле LibGraph.cpp, которая по множеству точек на плоскости

$$X = (x_0 \quad x_1 \quad \dots \quad x_N)^T, \quad Y = (y_0 \quad y_1 \quad \dots \quad y_N)^T$$

вычисляет координаты **кривой Безье**

$$XB = (x_0^b \quad x_1^b \quad \dots \quad x_M^b)^T, \quad YB = (y_0^b \quad y_1^b \quad \dots \quad y_M^b)^T,$$

M – число отрезков, на которые разбивается параметр t, $t \in [0,1]$

```
void CPlot2D::DrawLagr(CDC& dc)
{
 double dx = pi / 4;
 double xL = 0;
 double xH = pi;
 int N = (xH - xL) / dx;
```

```
 dx = 0.2;
 int NL = (xH - xL) / dx;
 CMatrix XL(NL + 1);
 CMatrix YL(NL + 1);

 for (int i = 0; i <= NL; i++)
 {
        XL(i) = xL + i * dx;
        YL(i) = Lagr(X, Y, XL(i), N + 1);
 }

 double xs, ys;
 int xw, yw;
 xs = XL(0); ys = YL(0);
 GetWindowCoords(xs, ys, xw, yw);  // координаты начальной точки графика в ОСК
 CPen MyPen(PenLine.PenStyle, PenLine.PenWidth, PenLine.PenColor);
 CPen* pOldPen = dc.SelectObject(&MyPen);
 dc.MoveTo(xw, yw);                                  // Перо в начальную точка для рисования графика
 for (int i = 1; i < XL.rows(); i++)
 {
        xs = XL(i); ys = YL(i);
        GetWindowCoords(xs, ys, xw, yw);  // координаты начальной точки графика с номером i в ОСК
        dc.LineTo(xw, yw);
 }
 dc.SelectObject(pOldPen);
}
```

Для четвертого задания построения кривых Безье в классе CChidView файл ChildView.h определяем:

```
// действия при выборе пункта меню
 afx_msg void OnBezier();
 afx_msg void OnBezier1();
 afx_msg void OnBezier2();
 afx_msg void OnBezier3();
 afx_msg void OnBezier4();
 afx_msg void OnLagr();
```

Реализация в ChildView.cpp

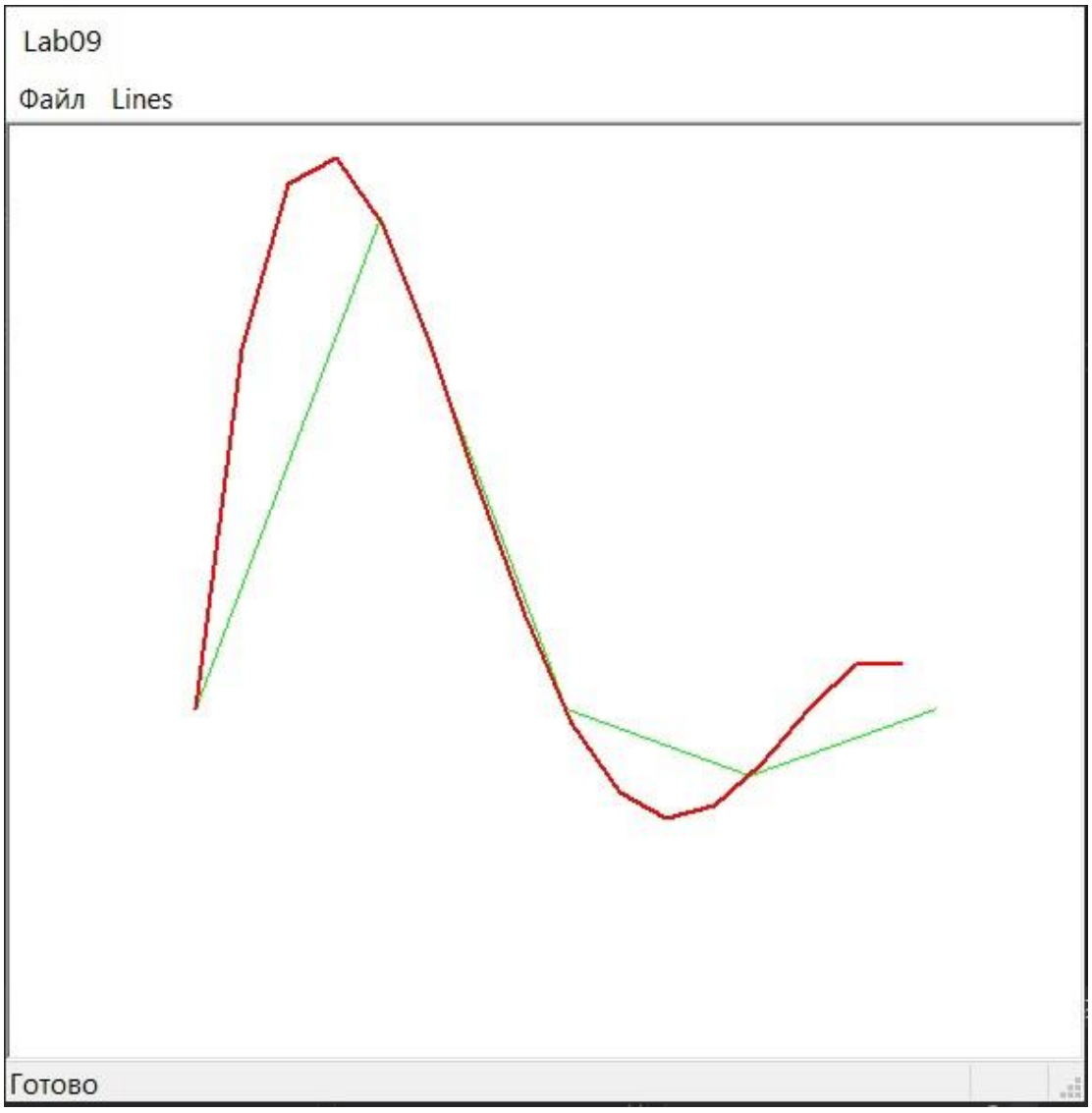
```
void CChildView::OnBezier()
{
 double dt = pi / 4;
 int N = 9;
 X.RedimMatrix(N);
 Y.RedimMatrix(N);
 for (int i = 0; i < N; i++)
 {
        X(i) = i * dt;
        Y(i) = sin(i*dt);
 }
 N_Bezier = 50;
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 2;
 this->Invalidate();
}


void CChildView::OnBezier1()
{
```

```
 double dt = pi / 3;
 int N = 4;
 X.RedimMatrix(N);
 Y.RedimMatrix(N);
 /*for (int i = 0; i < N; i++)
 {
        X(i) = i * dt;
        Y(i) = sin(i*dt);
 }*/
 X(0) = 0;
 Y(0) = 0;
 X(1) = 3 * dt;
 Y(1) = dt;
 X(2) = 5 * dt;
 Y(2) = dt;
 X(3) = 6 * dt;
 Y(3) = 0;
 N_Bezier = 50;
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 3;
 this->Invalidate();
}
void CChildView::OnBezier2()
{
 double dt = pi / 2;
 int N = 3;
 X.RedimMatrix(N);
 Y.RedimMatrix(N);
 /*for (int i = 0; i < N; i++)
 {
        X(i) = i * dt;
        Y(i) = sin(i*dt);
 }*/
 X(0) = 0;
 Y(0) = 0;
 X(1) = 1.5 * dt;
 Y(1) = 2;
 X(2) = 3*dt;
 Y(2) = 0;
 N_Bezier = 50;
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 4;
 this->Invalidate();
}
void CChildView::OnBezier3()
{
 double dt = pi / 2;
 int N = 5;
 X.RedimMatrix(N);
 Y.RedimMatrix(N);
 for (int i = 0; i < N; i++)
 {
        X(i) = i * dt;
        Y(i) = sin(i*dt);
 }
 N_Bezier = 50;
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 5;
 this->Invalidate();
```

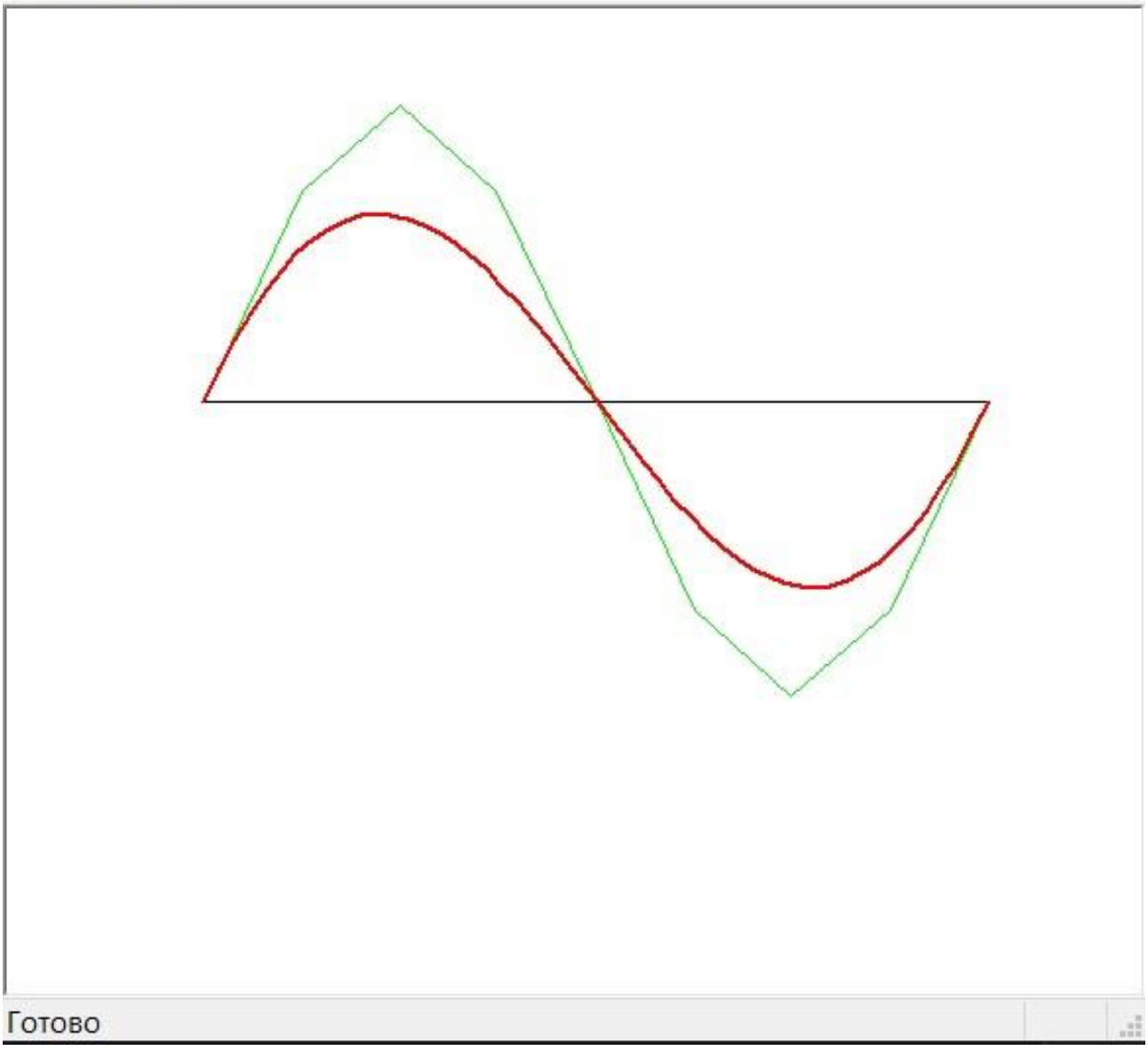
```
}
void CChildView::OnBezier4()
{
 double dt = pi / 4;
 int N = 4;
 X.RedimMatrix(N);
 Y.RedimMatrix(N);
 X(0) = 2*dt;
 Y(0) = 0;
 X(1) = 7 * dt;
 Y(1) = 3;
 X(2) = 0;
 Y(2) = 3;
 X(3) = 5 * dt;
 Y(3) = 0;
 N_Bezier = 50;
 RW.SetRect(100, 50, 500, 350);
 Graph.SetParams(X, Y, RW);
 Index = 6;
 this->Invalidate();
                                                    }
```

Результат работы:

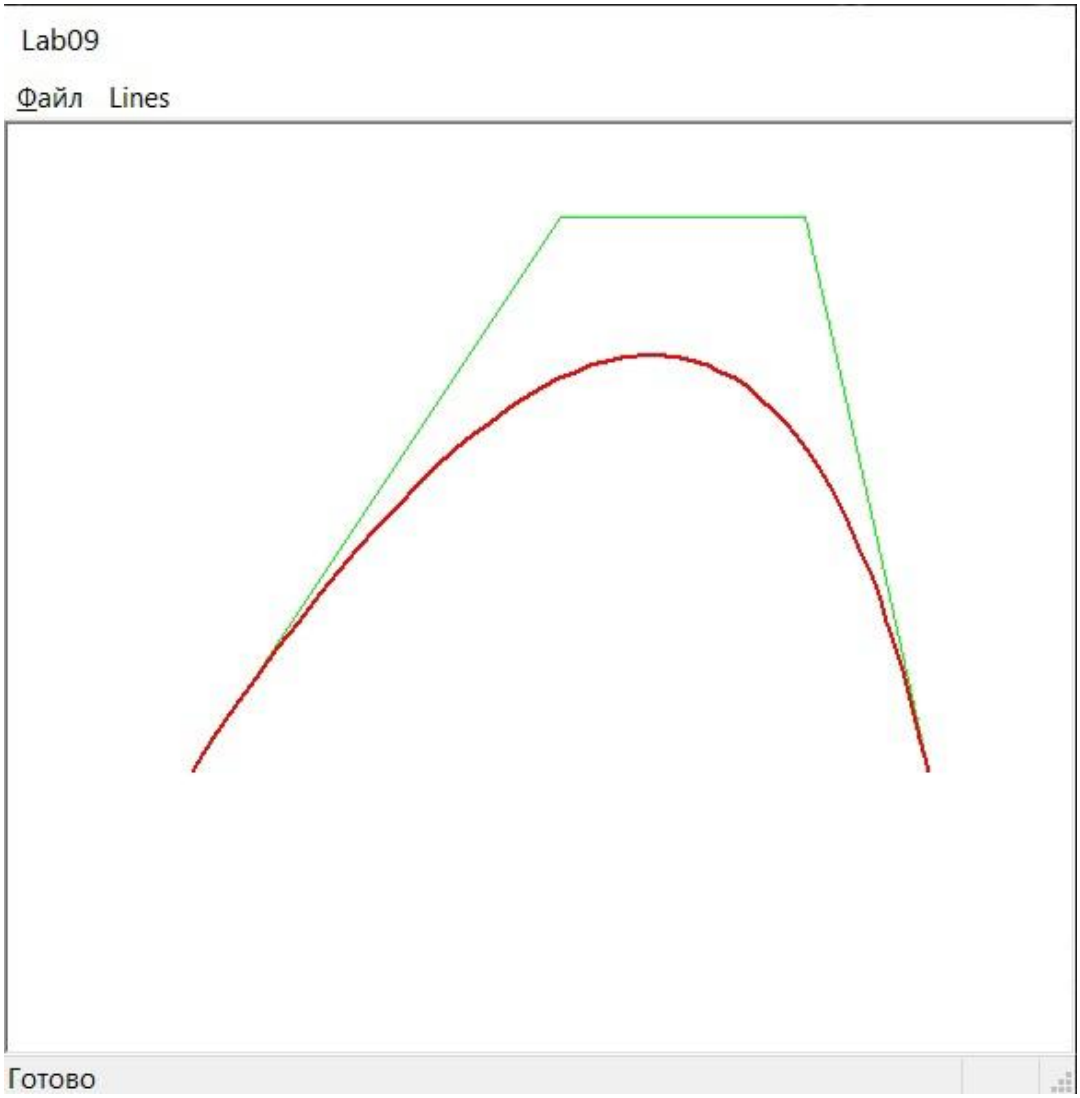


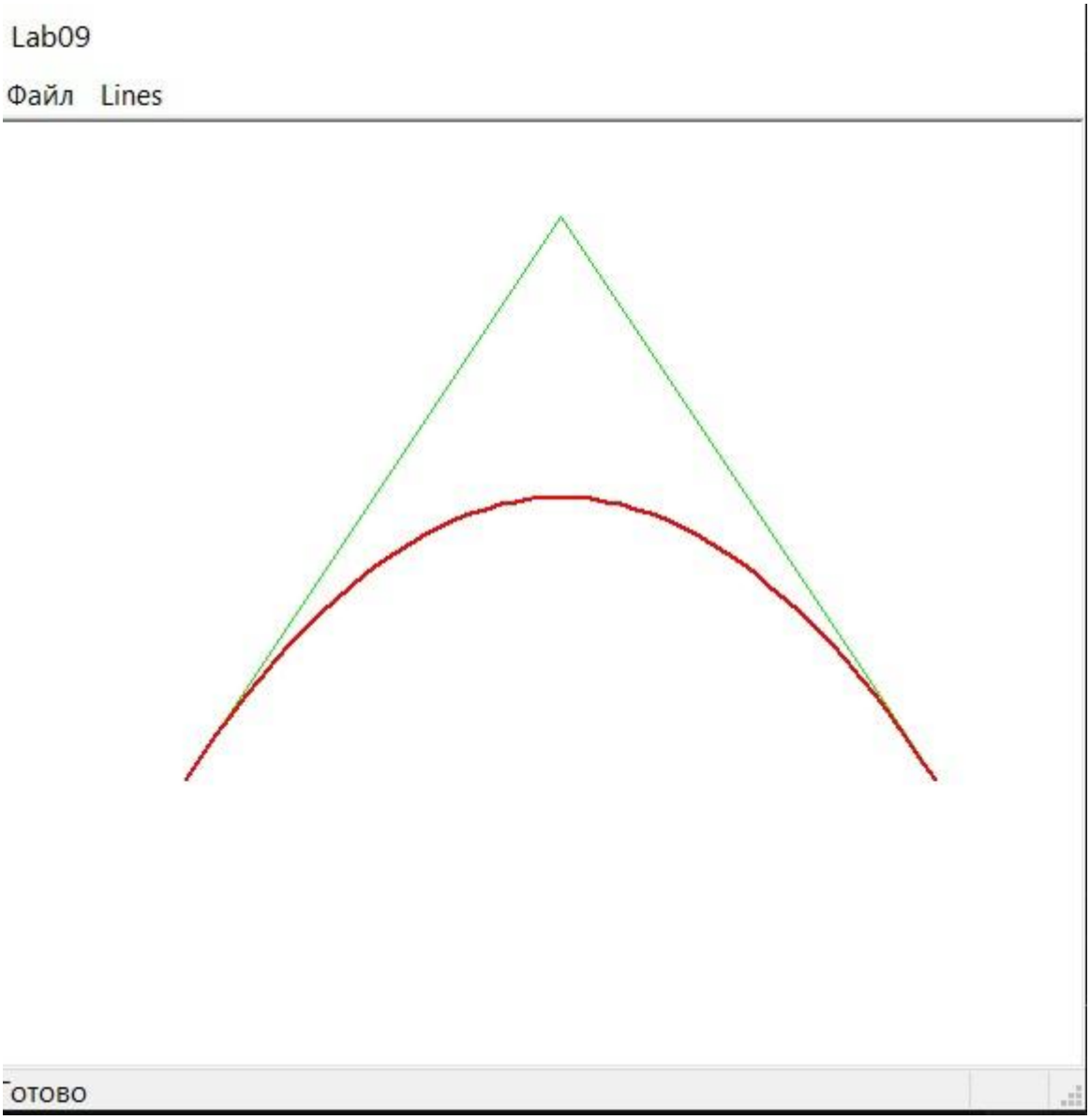
Lab09

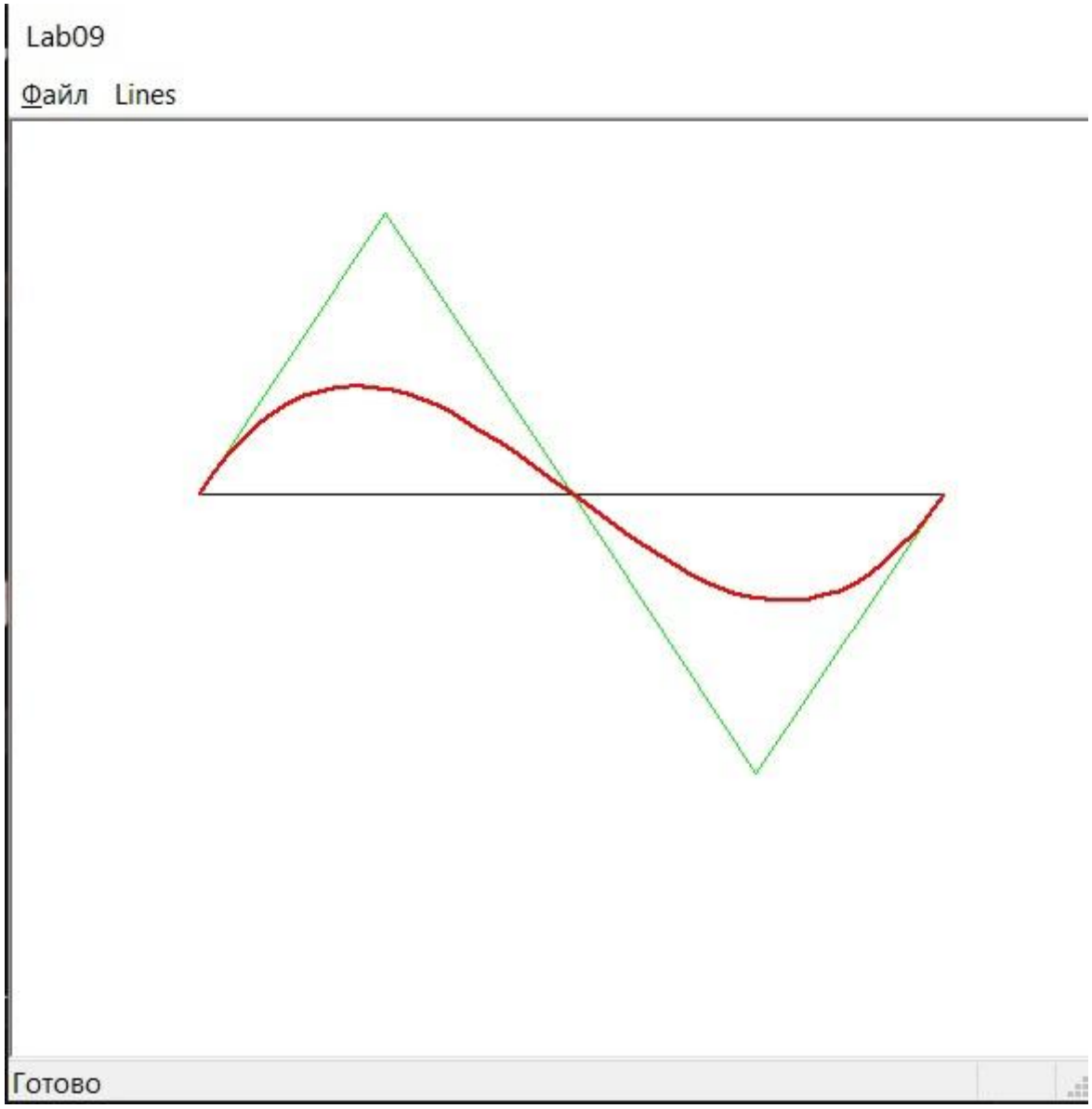
Файл Lines

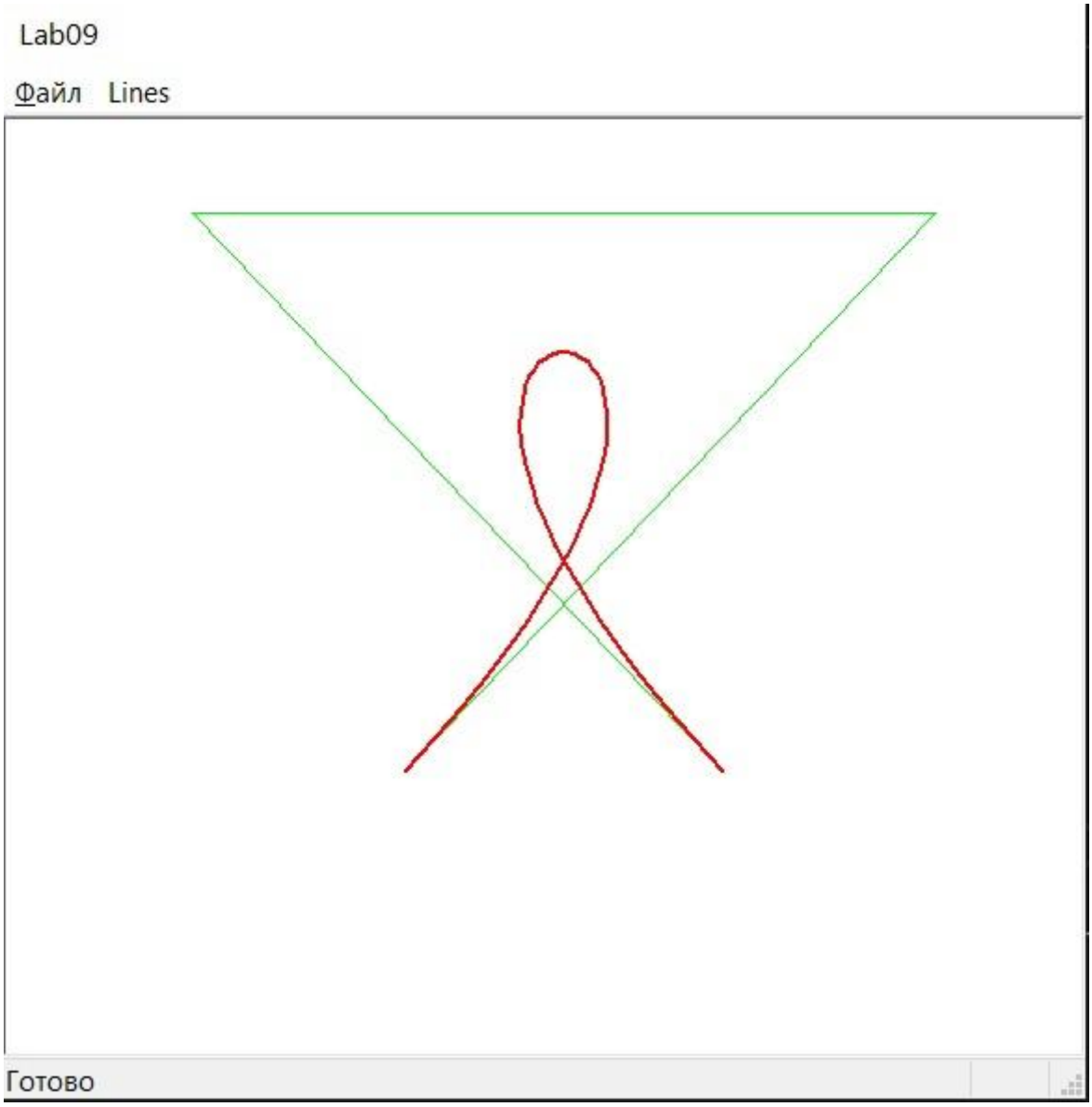


Готово


Lab09
Файл Lines
Готово


Lab09
Файл Lines
Готово


Lab09
Файл Lines
Готово



Литература

http://simenergy.ru/math-analysis/digital-processing/76-lagrange-polynomial
https://ru.wikipedia.org/wiki/Интерполяционный_многочлен_Лагранжа
https://portal.tpu.ru/SHARED/m/MOE/Ucheba/Tab3/Tab/LK_Интерполирование_М_Лагранжа.pdf